What Is Claimed:

1. A virtualization switch for performing a plurality of virtualization services within a data path said virtualization switch comprises at least:

a network interface (NI);

an iSCSI module;

a target manager (TM);

a volume manager (VM);

a data transfer arbiter (DTA);

a device manager (DM);

a plurality of input ports to receive incoming packets from a network; and,

a plurality of output ports to communicate with plurality of storage devices.

2. The virtualization switch of claim 1, wherein said virtualization switch is capable of operating in at least one of: storage area network (SAN), network attached storage (NAS).

3. The virtualization switch of claim 1, wherein said data path is established between a host and said storage devices.

4. The virtualization switch of claim 1, wherein said virtualization services comprise at least one of: mirroring, remote mirroring over a slow link, snapshot, data replication, striping, concatenation, periodic local and remote backup, restore.

5. The virtualization switch of claim 1, wherein said network is at least one of: local area network (LAN), wide area network (WAN), geographically distributed network.

6. The virtualization switch of claim 1, wherein said storage device is at least one of: tape drive, optical drive, disk, sub-disk, redundant array of inexpensive disks (RAID).

7. The virtualization switch of claim 1, wherein said input ports are capable of carrying packets in accordance with a transport protocol.

8. The virtualization switch of claim 7, wherein said transport protocol is at least one of: Fiber Cannel (FC), parallel small computer system interface (SCSI) , internet small computer system interface (iSCSI), transmission control protocol (TCP)/ internet protocol (IP), Infiniband.

9. The virtualization switch of claim 1, wherein said output ports are capable of carrying packets in accordance with a transport protocol.

10. The virtualization switch of claim 9, wherein said transport protocol is at least one of: Fiber Cannel (FC), parallel SCSI, iSCSI, TCP/IP, Infiniband.

11. The virtualization switch of claim 1, wherein said NI further comprises a TCP/IP stack for the purpose of accelerating TCP/IP packets processing.

12. The virtualization switch of claim 1, wherein said iSCSI module further comprises an iSCSI stack for the purpose of handling an iSCSI protocol.

13. The virtualization switch of claim 1, wherein said TM comprises instructions for the purpose of:

handling incoming logic commands; and,

scheduling the execution of said incoming logic commands.

14. The virtualization switch of claim 13, wherein said logic command refers to a virtual volume and a virtual address space.

15. The virtualization switch of claim 13, wherein said logic command is at least SCSI command.

16. The virtualization switch of claim 13, wherein said TM further comprises a plurality of host-logical unit (LU) queues, wherein each of said host-LU queue contains said logic commands requested to be executed by said host on said LU.

17. The virtualization switch of claim 16, wherein said LU comprises a plurality of contiguous partitions of storage space on said storage device.

18. The virtualization switch of claim 1, wherein said DTA is capable of handling data transfer between said storage devices and hosts.

19. The virtualization switch of claim 1, wherein said VM is capable of translating a logic command to a list of physical commands.

20. The virtualization switch of claim 19, wherein each of said physical commands includes at least: a physical address of a single storage device.

21. The virtualization switch of claim 18, wherein said physical commands are constructed in a data structure, said data structure defines the relations between said physical commands.

22. The virtualization switch of claim 21, wherein said data structure comprises at least one of: alternative command link, pointer to said storage device.

23. The virtualization switch of claim 22, wherein said alternative command link links between at least two physical commands that can be executed in parallel.

24. The virtualization switch of claim 21, wherein said VM further comprises a mapping schema uses for translating said logic command to said list of said physical commands.

25. The virtualization switch of claim 24, wherein said mapping schema defines relations between virtual volumes, logical units (LUs), and said storage devices.

26. The virtualization switch of claim 25, wherein said virtual volume is at least one of: concatenation volume, stripe volume, mirrored volume, simple volume, snapshot volume.

27. The virtualization switch of claim 1, wherein said DM comprises at least:
a list of target paths; and,
a list of LU paths associated with each of said target paths.

28. The virtualization switch of claim 27, wherein each of said target paths defines a connection between said virtualization switch and one of said storage devices, via one of said output ports.

29. The virtualization switch of claim 27, wherein said DM further comprises a plurality of storage drivers for communicating with said plurality of output ports.

30. The virtualization switch of claim 1, wherein said virtualization switch further provides a bridge mechanism for transferring data without performing said virtualization services.

31. The virtualization switch of claim 1, wherein said virtualization switch is further capable of reporting on error generated by virtual volumes.

32. A method for performing a plurality virtualization services, said method being further operative to perform said virtualization services within a data path, said method comprises the steps of:

a) receiving a logic command to be performed on at least one virtual volume, said logic command including at least a virtual address;

d) scheduling said logic command for execution;

c) translating, in one pass, said logic command to a list of physical commands, wherein each of said physical commands is targeted to a different storage device;

d) determining the amount of data to be transferred via a network; and,

e) executing said physical commands on said storage devices.

33. The method of claim 32, wherein said virtualization services comprise at least one of: mirroring, remote mirroring over a slow link, snapshot, data replication, striping, concatenation, periodic local and remote backup, restore.

34. The method of claim 32, wherein said data path is established between a host and said storage devices.

35. The method of claim 32, wherein said storage device is at least one of: a tape drive, optical drive, disk, sub-disk, redundant array of inexpensive disks (RAID).

36. The method of claim 32, wherein said logic command is at least a SCSI command.

37. The method of claim 36, the following steps comprise receiving said logic command:

a) initiating an iSCSI session with an initiator host;

b) receiving said logic command from said initiator host;

c) parsing said logic command to determine at least said virtual address and said logic command's type;

d) performing a check to determine if said logic command is valid;

e) generating a response command if said logic command is invalid, otherwise, adding said logic command to a host-LU queue; and,

f) generating a data transfer request.

38. The method of claim 37, wherein the following steps further comprise initiating said iSCSI session:

a) determining if said initiator host is authorized to send said logic command; and,

b) denying said logic command from said initiator host, if said initiator host is unauthorized.

39. The method of claim 37, wherein said response command comprises an iSCSI service response code indicating the type of a generated error.

40. The method of claim 37, wherein said host-LU queue comprises logic commands requested to be executed by said host on said LU.

41. The method of claim 37, wherein scheduling said logic command for execution further comprises the step of: selecting said logic command to be executed from said host-LU queue.

42. The method of claim 41, wherein the selection is performed using at least one of the following selection algorithms: recently used, round robin, weighted round robin, random, least loaded LU.

43. The method of claim 37, wherein said command type is a read command.

44. The method of claim 43, wherein said amount of data to be transferred is determined by an available space parameter.

45. The method of claim 44, wherein said available space parameter defines the number of data bytes to be sent to the host.

46. The method of claim 44, wherein the following steps comprise executing said physical commands on said storage devices:

a) accessing a storage device using a physical address;

b) retrieving from said accessed storage device the number of bytes designated in said available space parameter;

c) sending the retrieved data to said host; and,

d) repeating said steps a) through d) until all requested data is read from said storage devices.

47. The method of claim 46, wherein said physical commands are executed in parallel.

48. The method of claim37, wherein said command type is a write command.

49. The method of claim 48, wherein said amount of data to be transferred is determined by a check-point list.

50. The method of claim 49, wherein said check-point list defines how data should be sent from an initiator host to said storage devices.

51. The method of claim 50, wherein said check-point list comprises a linked list of data chunks.

52. The method of claim 51, wherein the following steps comprise executing said physical commands on said storage devices:

a) filling at least one data chunk with said data retrieved from the network;

b) accessing said storage device using a physical address;

c) writing said data chunk to said accessed storage device; and,

d) repeating said steps a) through d) for all data chunks in said check-point list.

53. The method of claim 52, wherein said physical commands are executed in parallel.

54. The method of claim 32, wherein said physical commands are constructed in a data structure.

55. The method of claim 54, wherein said data structure further includes at least one of: an alternative command link, a pointer to said storage device.

56. The method of claim 55, wherein said alternative command link links between at least two physical commands that can be executed in parallel.

57. The method of claim 32, wherein translating said logic command to said list of physical commands is performed using a mapping schema.

58. The method of claim 57, wherein said mapping schema defines relations between virtual volumes, logical units (LUs), and said storage devices.

59. The method of claim 32, wherein upon completing the execution of said physical commands further comprises the steps of:

a) removing said logic command from the host-LU queue;

b) sending to the initiator host a response command, said response command signals the end of execution.

60. The method of claim 32, wherein said method is further capable to perform operations on said virtual volumes that do not require any data transfer.

61. The method of claim 32, wherein said method is further capable of reporting on errors generated by virtual volumes.

62. A computer executable code for performing a plurality virtualization services, said computer executable code being further operative to perform said virtualization services within a data path, said code comprises the steps of:

a) receiving a logic command to be performed on at least one virtual volume, said logic command including at least a virtual address;

d) scheduling said logic command for execution;

c) translating, in one pass, said logic command to a list of physical commands, wherein each of said physical commands is targeted to a different storage device;

d) determining the amount of data to be transferred via a network; and,

e) executing said physical commands on said storage devices.

63. The computer executable code of claim 62, wherein said virtualization services comprise at least one of: mirroring, remote mirroring over a slow link, snapshot, data replication, striping, concatenation, periodic local and remote backup, and restore.

64. The computer executable code of claim 62, wherein said data path is established between a host and said storage devices.

65. The computer executable code of claim 62, wherein said storage device is at least one of: tape drive, optical drive, disk, sub-disk, redundant array of inexpensive disks (RAID).

66. The computer executable code of claim 62, wherein said logic command is at least a SCSI command.

67. The computer executable code of claim 66, the following steps comprise receiving said logic command:

a) initiating an iSCSI session with an initiator host;

b) receiving said logic command from said initiator host;

c) parsing said logic command to determine at least said virtual address and said logic command's type;

d) performing a check to determine if said logic command is valid,

e) generating a response command if said logic command is invalid, otherwise, adding said logic command to a host-LU queue; and,

f) generating a data transfer request.

68. The computer executable code of claim 67, wherein the following steps further comprise initiating said iSCSI session:

a) determining if said initiator host is authorized to send said logic command; and,

b) denying said logic command from said initiator host, if said initiator host is unauthorized.

69. The computer executable code of claim 67, wherein said response command comprises an iSCSI service response code indicating the type of error.

70. The computer executable code of claim 67, wherein said host-LU queue comprises logic commands requested to be executed by said host on said LU.

71. The computer executable code of claim 67, wherein scheduling said logic command for execution further comprises the step of: selecting said logic command to be executed from said host-LU queue.

72. The computer executable code of claim 71, wherein the selection is performed using at least one of the following selection algorithms: recently used, round robin, weighted round robin, random, least loaded LU.

73. The computer executable code of claim 67, wherein said command type is a read command.

74. The computer executable code of claim 73, wherein said amount of data to be transferred is determined by an available space parameter.

75. The computer executable code of claim 74, wherein said available space parameter defines the number of data bytes to be sent to the initiator host.

76. The computer executable code of claim 73, wherein the following steps comprise executing said physical commands on said storage devices:

a) accessing a storage device using a physical address;

b) retrieving from said accessed storage device the number of bytes designated in said available space parameter;

c) sending the retrieved data to said host; and,

d) repeating said steps a) through d) until all requested data is read from said storage devices.

77. The computer executable code of claim 76, wherein said physical commands are executed in parallel.

78. The computer executable code of claim 67, wherein said command type is a write command.

79. The computer executable code of claim 78, wherein said amount of data to be transferred is determined by a check-point list.

80. The computer executable code of claim 79, wherein said a check-point list defines how data should be sent from the initiator host to said storage devices.

81. The computer executable code of claim 80, wherein said check-point list comprises a linked list of data chunks.

82. The computer executable code of claim 81, wherein the following steps comprise executing said physical commands on said storage devices:

a) filling at least one data chunk with said data retrieved from the network;

b) accessing said storage device using a physical address;

c) writing said data chunk to said accessed storage device; and,

d) repeating said steps a) through d) for all data chunks in said check-point list.

83. The computer executable code of claim 82, wherein said physical commands are executed in parallel.

84. The computer executable code of claim 83, wherein said physical commands are constructed in a data structure.

85. The computer executable code of claim 84, wherein said data structure further includes at least one of: an alternative command link, a pointer to said storage device.

86. The computer executable code of claim 85, wherein said alternative command link links between at least two physical commands that can be executed in parallel.

87. The computer executable code of claim 62, wherein translating said logic command to said list of physical commands is performed using a mapping schema.

88. The computer executable code of claim 87, wherein said mapping schema defines relations between virtual volumes, logical units (LUs), and said storage devices.

89. The computer executable code of claim 62, wherein upon completing the execution of said physical commands further comprises the steps of:

a) removing said logic command from the host-LU queue;

b) sending to the initiator host a response command, said response command signals the end of execution.

90. The computer executable code of claim 62, wherein said code is further capable to perform operations on said virtual volumes that do not require any data transfer.

91. The computer executable code of claim 62, wherein said method is further capable of reporting on errors generated by virtual volumes.